How Objects Store ?

Stack (1 MB windows and Linux 8BM , Small fixed size from (1 MB → 8 MB))
| Heap (Big storage dynamic , On modern 64-bit systems, this can be in the gigabytes)

Value types (store the value its self in memory with specific part)

- numeric , Boolean, char , struct

| Reference types (store its address in part and its actual data in stored in another part of memeory)

- array
- string
- classes
- objects

▼ Reference :

References: The variable doesn't store the object directly. Instead, it stores the memory address where the object is located. This is why multiple variables can reference the same object; they all point to the same memory location.

- **When you create object using `new` : this object with properties and methods stored in heap (dynamic allocation) , but can not reach this object if i don’t have address or reference to it ,**
  so you store the reference of that object in stack and through that variable that assign to object to store its address you can access the actual data objects that stored in heap

> The stack keeps track of the reference, but the actual object's data is in the heap

Example :

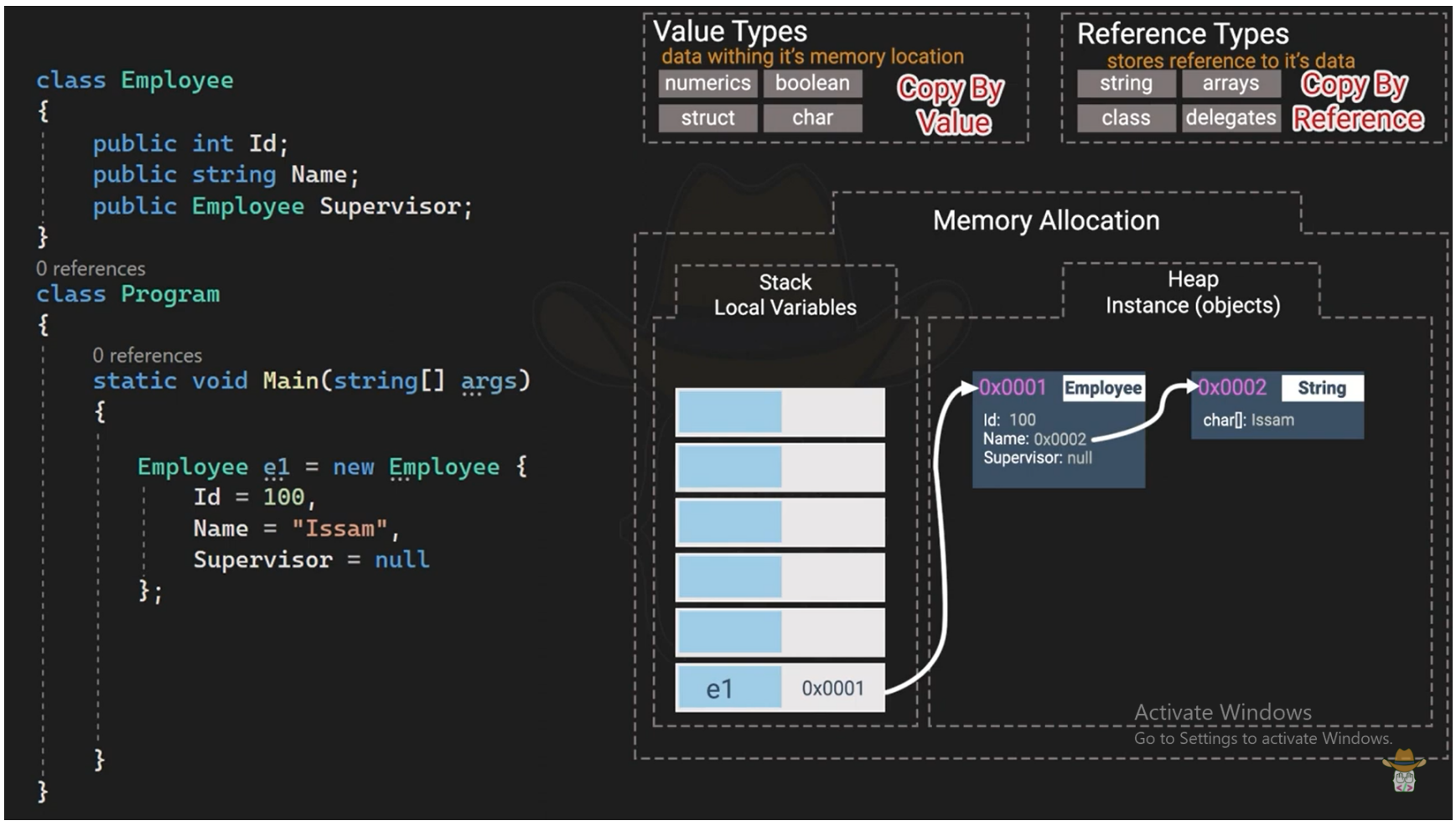
```
class Car {
    public $color;
    public function __construct($color) {
        $this->color = $color;
    }
}

$car1 = new Car("red");
$car2 = $car1;
```

- **variable that store the reference of the object created through `new` is `$car1`**

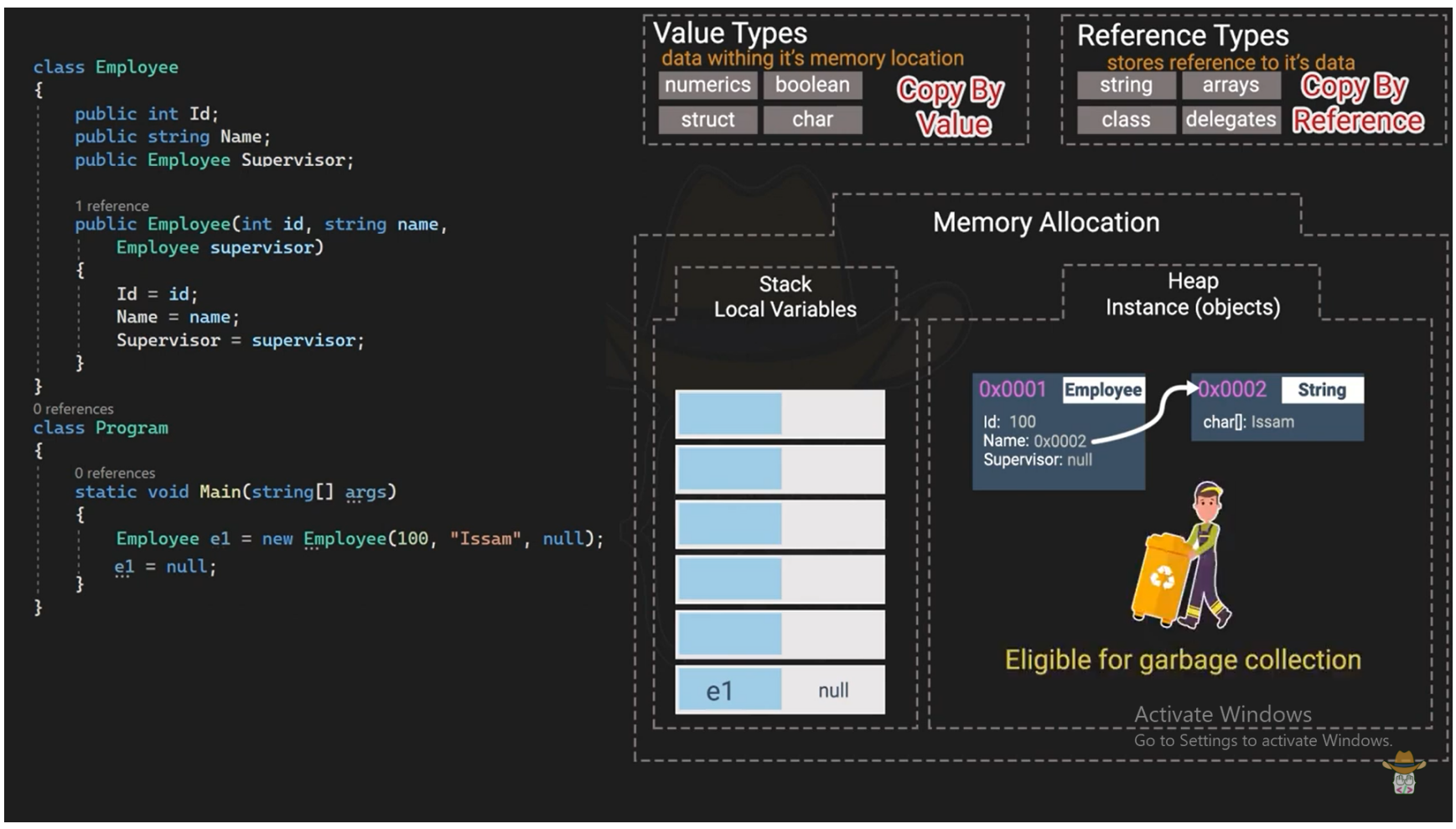
`$car2` is then assigned the same reference as `$car1`, meaning both `$car1` and `$car2` point to the same object in memory. Changing the colour of the car through one reference will reflect in the other.

> so the `$car1` and `$car2` points to the same object in heap memory because `$car2` takes , stores , assigns the same reference of
>
> `$car1` so you can access through one of them as they points to the same actual data

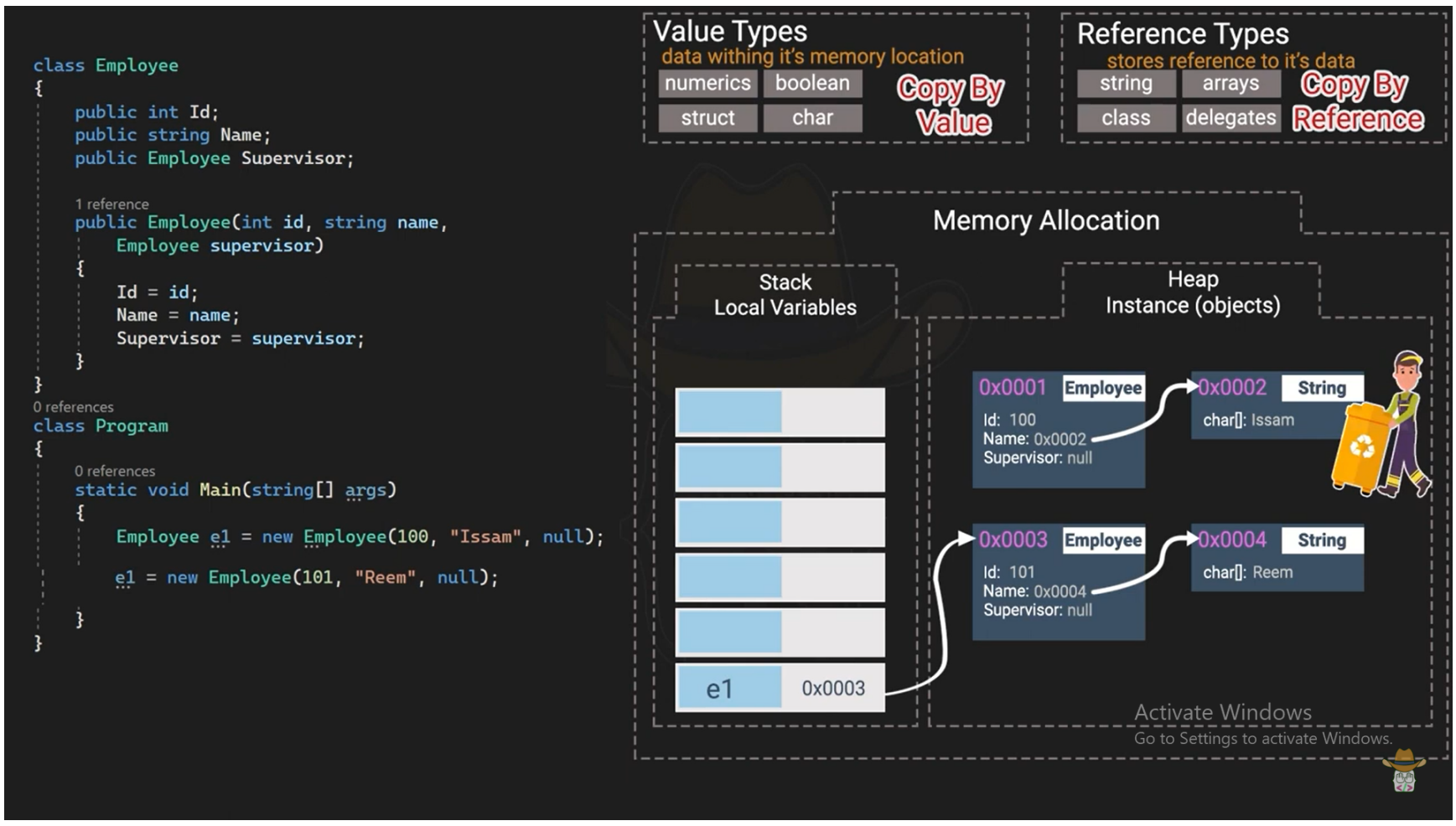


As we know : you can initialize properties through constructor or assign values to int directly if public

What if I created an object and then assign `null` to variable reference for that object :



What if I assign the same variable `e1` to another object , so i can not access the previous object : بقا ملوش أهل يسألو عليه



Here as we said : I assign the reference not address to any modifications or changes will be the same in both